REMARKS

This application has been carefully considered in connection with the Office Action dated March 13, 2009. Reconsideration and allowance are respectfully requested in view of the following.

Summary of Rejections

Claims 1-33 were pending at the time of the Office Action.

Claims 1-33 were rejected under 35 USC § 103.

Summary of Response

Claims 1, 9-12, 28, and 30 are currently amended.

Claims 2-3, 23, and 31 were previously presented.

Claims 4-8, 13-22, 24-27, and 32-33 remain as originally submitted.

Claim 29 is canceled.

Remarks and Arguments are provided below.

Summary of Claims Pending

Claims 1-28 and 30-33 are currently pending following this response.

Applicant Initiated Interview

Applicants thank Examiner Shanto Abedin for his time and consideration of the arguments presented in the interview on June 2, 2009. In the interview Applicants noted that the claims of the pending application provided a lightweight solution for providing application-to-application enterprise security where the applications reside on different platforms by using a platform and application independent token. Applicants also noted that the client application and the server application are distinct from each other and operate on different platforms. The Examiner agreed that it appeared that providing application to application enterprise security between distinct applications on different platforms might not be taught by the cited portions of the prior art, but indicated that a new search would be required. A detailed discussion of the differences between the applied art and the claim limitations follows.

In the interest of advancing prosecution, the claims have been amended herein as suggested by Examiner Abedin.

Response to Rejections

The systems and methods of the pending application includes a security application program interface, an authentication authority, a store maintaining data, an application program interface, and a server application. The system and methods of the pending application provides a lightweight solution to enforcing security for communications between disparate applications executing on different platforms that does not require each application to be reprogrammed to include an application interface for each individual application with which it must communicate. The system and methods of

the pending application provides for platform and application independent tokens to be passed among disparate applications residing on different platforms so that security information can automatically be included with each call from one application to another. This mitigates the need for an application to be authenticated and authorized every time it sends a message to another application. Thus, in contrast with services where a security context remains present on a server, the claims of the pending application provided security for applications in which there is no permanent context or session. Instead, a new context is created with new invocations from one application to another.

Also, rather than security information being converted from the format of one platform to the format of another, security information is passed between applications in the form of a token with a string data type that makes the token platform and application independent. In other words, since a string is a primitive data type, it can be recognized by a large number of applications and interfaces, meaning it can be sent over multiple services. Furthermore, making the token a string makes it platform and technology independent because the token has no header and therefore no application-specific header configuration. Thus, the claims of the pending application provide methods and systems that provide a lightweight solution for enforcing security between disparate applications on different platforms that mitigates the need for a converter between every possible combination of different applications that would communicate with each other.

With regard to the art rejections, the Office Action has cited Upton, U.S. Pub. No. 2003/0097574 A1 (“Upton”); Bhat et al., U.S. Pub. No. 2003/0200465 A1 (“Bhat”); and Bhatia et al., U.S. Patent No. 7,249,375 B2 (”Bhatia”). Upton, Bhat, and Bhatia, alone or in combination, do not disclose, teach, or suggest a token containing user credentials

encoded as a platform and application independent string data type and providing the token to a server application, as claimed. Making the token a string data type makes it platform and application independent because the token has no header and, therefore, no application-specific header configuration. Thus, notably, the tokens found in the pending disclosure permit the authentication of users in a heterogeneous computing environment. This feature of the claimed token mitigates the need to convert security information from the format of one platform to the format of another.

Upton relates to systems and methods for integration adapter security. Bhat and Bhatia both relate to systems and methods for single sign-on to Web-based applications. Bhat discloses a token that is sent to multiple applications, but nothing in Bhat teaches or suggests that these applications require different formatting. In fact, it appears that the opposite is true: that all of the plurality of applications for which the token is recognized share a common formatting requirement for messages and other communications.

These distinctions, as well as others, will be discussed in greater detail in the analysis of the present claims that follows.

Response to Rejections under Section 103

Claim 1:

Claim 1 was rejected under 35 USC § 103(a) as being unpatentable over Upton, U.S. Pub. No. 2003/0097574 A1 (hereinafter, "Upton") in view of Bhat et al., U.S. Pub. No. 2003/0200465 A1 (hereinafter, "Bhat") further in view of Bhatia et al., U.S. Patent No. 7,249,375 B2 ("Bhatia").

I. None of the applied art provides for a system to provide application-to-application enterprise security for different applications on different platforms where there is no continuing context or session and a new context is created with new invocations from one of the applications to another.

Claim 1, as amended, recites "[a] system to provide application-to-application enterprise security for different applications on different platforms where there is no continuing context or session and a new context is created with new invocations from one of the applications to another." Applicants respectfully submit that neither Upton, Bhat, nor Bhatia teach or suggest application-to-application enterprise security for different applications on different platforms where there is no continuing context or session. Claim 1 has been amended to make it clear that the client application and the distinct server application are different applications from each other and that they each are on separate distinct and different platforms. For example, claim 1 has been amended to recite, "a distinct server application on a second platform to receive the token from the application program interface, the server application communicating with the authentication authority to validate the token to enable the client application to use services of the server application, wherein a new context is created with an invocation

of the distinct server application by the client application." The use of the platform and independent token to provide application-to-application enterprise security between different and distinct applications on different platforms provides a lightweight solution that does not require an application to be reprogrammed with a separate application program interface for each of the other applications with which it may communicate with and require enterprise security.

Upton teaches the use of "user identity tokens," but does not teach or suggest "wherein a new context is created with an invocation of the distinct server application by the client application," as recited in claim 1. Upton also does not teach that the client application and the distinct server application are on different platforms from each other. Bhat teaches a single sign-on system to web-based applications by issuing an identifying token to the user. Bhatia also teaches an SSO token that is application dependent since SSO server 106 issues tokens in multiple formats based upon the capabilities of the target system for operation in a heterogeneous environment. Furthermore, none of the applied art teaches "wherein a new context is created with an invocation of the distinct server application by the client application" as required by claim 1.

II. Upton, Bhat, and Bhatia do not disclose, teach or suggest wherein the token contains user credentials encoded as a platform and application independent string data type.

Claim 1 (as previously presented) recites, in part, "wherein the token contains user credentials encoded as a platform and application **independent** string data type..." (emphasis added).

With respect to claim 1, the Office Action states, in pertinent part:

> ***Regarding claim 1,*** Upton discloses a system to provide application-to-application enterprise security, the system comprising:
>
> …an authentication authority (Par 0063-0065, 0104, 0128, 0145,; SAS, or JAAS, or authentication/ authorization SPI) receiving the security credential (Par 0061-0069; credentials; security-pinciple; ra.xml file) from the security application program interface, the authentication authority further operable to communicate the token to the security application program interface where the security credential is valid, wherein the token contains user credentials encoded as a platform and application independent string data type (fig 4; Par 0063-0069, 0104, 0114, 0130, 0150; service provider interface/ SPI; checking public/ password type, or generic token type credentials, or security-principal map element).

Office Action dated March 13, 2009, Pages 2-3.

Contrary to the assertions in the Office Action, Upton does not disclose, teach or suggest "wherein the token contains user credentials encoded as a platform and application independent string data type," as recited in claim 1. The platform independent token of the pending claims provides for application-to-application enterprise security between distinct applications residing on different platforms. Furthermore, the use of the token to provide application-to-application enterprise security for distinct applications residing on different platforms provides a lightweight solution that does not require significant reprogramming of each application to include an application program interface for each application with which it may interface. In rejecting claim 1, the Office Action relied on paragraph [0150] in Upton to read on the claimed token. Paragraph [0150] of Upton states:

> [0150] A Principal→Credential Mapping (JAAS Login-Module) SPI can be based on the JAAS Login Module, and can be used to map principal identity when cross security domain policy or technology boundaries. The responsibilities of the Principal Mapping SPI is based on the Subject provided, and can be used to add public credentials with appropriate information to subject, such as password credentials for username/password, and generic credential for token-type credentials. (Underlining added for emphasis.)

The disclosure in Upton of "generic credential for token-type credentials" is not the token recited in claim 1 of the pending application. As pointed out in Applicant's previous response, a text search of Upton for the string "generic" reveals that the word is used only twice: once in paragraph 0150 cited above and once in paragraph 0065. Upton, paragraph 0065 recites that "the resource adapters can support: password credentials and generic credentials." It is clear from the context that "generic credential" in no way refers to "platform and application independent" credentials, but is merely contrasting a "generic credential" with a "password credential". Thus, "generic credentials" are merely non-password credentials.

Also, the system of Upton is based on mapping credentials from a user to a resource. For example, paragraph 0087 of Upton recites:

> The "EIS Sign-on" section of the J2EE Connector Specification, Version 1.0 Final Release identifies a number of possible options for defining a Resource Principal on whose behalf the sign-on is being performed. Previous implementations implemented the Security Principal Map option identified in the specification. Under this option, a resource principal is determined by mapping from the identity of the initiating/caller principal for the invoking component. The resultant resource principal does not inherit the identity

> or security attributes of the principal that it is mapped from, but instead gets its identity and security attributes (password) based upon the defined mapping.

However, this method is contrary to the elements of claim 1 which do not require a mapping of a credential from one application to another. In fact, if a token is **platform** and application independent, there is no need to map the token or credential from one application to another. Mapping a token from one application to another is almost equivalent to reprogramming each application to have an application program interface for each program that it may interact with – which is one of the problems solved by the disclosure of the pending application. To better illustrate this and better understand the contrast between Upton and claim 1, paragraphs 0027 and 0028 of the pending application are provided below which state:

> **[0027]** The present system allows tokens to be passed among disparate applications so that security information can automatically be included with each call from one application to another. This eliminates the need for conversion of security information in message headers between the data format of the applications. It also eliminates the need for an application to be authenticated and authorized every time it sends a message to another application. In contrast with services where a security context remains present on a server, in embodiments of the invention there is no permanent context or session. Instead, a context is created with every invocation from one application to another.
>
> **[0028]** Rather than security information being converted from the format of one platform to the format of another, security information is passed between applications in the form of a token with a string data type. Since a string is a primitive data type, it can be recognized by a large number of applications and interfaces, meaning it can be sent over multiple services such as J2EE, CORBA, and

> IBM's MQSeries. Making the token a string makes it platform and technology independent because the token has no header and therefore no application-specific header configuration.

Mapping credential information as Upton teaches is equivalent to conversion of security information and is different from the token of claim 1 "wherein the token contains user credentials encoded as a platform and application independent string data type."

Additionally, nothing in this passage discloses that the "token contains user credentials encoded as a platform and application independent string data type" (emphasis added). In fact, the Office Action is silent with regard to the claimed "string data type." Upton does not disclose the data type for the credentials and does not suggest that the data type may be a "string data type." A text search of Upton for the term "string" produced six results, but none of these instances of the term "string" related to the token.

The Office Action also relied on the following disclosure in Upton to read on the same element of claim 1:

> [0104] For many systems, it may be desirable to include support for the Java 2 Enterprise Edition (J2EE) specification and interoperability therewith. These J2EE specification features include the Common Secure Interoperability (CSI) protocol, user identity tokens, the Stateless Authentication Service (SAS) protocol, support for propagation of security credentials across machine, cluster, and/or domain boundaries, control of propagation of identity based on policy, enhanced support for virtual host/sites, the ability to generate a user identity scoped to domain, and host/site specific security policies. (Underlining added for emphasis.)

As shown in the above-cited paragraph, Upton teaches the use of "user identity tokens" and does not teach or suggest "wherein the token contains user credentials encoded as a platform and application independent string data type," as recited in claim 1.

Bhat teaches a single sign-on system to web-based applications by issuing an identifying token to the user. Bhat is silent regarding a "platform and application independent" token. Bhat is also silent regarding a string data type. A text search of Bhat for the string "string" produced no results. Furthermore, Bhat appears not to recognize the problem of generating a token that is platform and application independent such that different tokens are not required for applications that do not share the same formatting requirements. Bhat discloses a token that is sent to multiple applications, but nothing in Bhat teaches or suggests that these applications require different formatting. In fact, it appears that the opposite is true: that all of the plurality of applications for which the token is recognized share a common formatting requirement for messages and other communications. In fact, the teachings of Bhat are similar to that disclosed in the background section of the pending application in paragraphs [0005]-[0006]. One problem of this approach is that the creation and passing of tokens is typically handled by proprietary, off-the-shelf authentication and authorization products that are **specific** to the applications in question and is **not** application and platform independent. (See, Application, ¶ [0007]).

Bhatia also teaches the use of a Single Sign On (SSO) token that provides a listener mechanism for applications that need notification when the SSO token expires. The Office Action alleges that Bhatia teaches a token containing user credentials

encoded as a platform and application independent string data type referencing Figure 3 and col. 3, starting at line 16 of Bhatia. Applicant respectfully disagrees with this assertion and respectfully submits that Bhatia does not disclose, teach or suggest "wherein the token contains user credentials encoded as a platform and application independent string data type," as recited in claim 1. In fact, Bhatia teaches the opposite. Bhatia teaches:

> SSO server 106 can issue tokens in multiple formats based upon the capabilities of the target system. Hence, SSO server 106 is able to do end-to-end identity propagation in a heterogeneous environment. Note that SSO server 106 can determine the type of target system-whether the target system is SSO partner application 108, RDBMS application 110, or third-party application 112 and in the case of third-party application 112 whether a user/password or standards based token is required. Based upon the target type, the appropriate token is issued.

(Bhatia, col. 3, lines 33-42)

Thus, rather than being platform, and application independent, the SSO token of Bhatia is application dependent since SSO server 106 issues tokens in multiple formats based upon the capabilities of the target system for operation in a heterogeneous environment. Therefore, according to the disclosure of Bhatia, a new token must be generated for each different platform and/or application that the user desires communication. Furthermore, nothing in Bhatia indicates that the token is a "string data type" as required by claim 1 of the pending application. A text search of Bhatia for the string "string" produced no results

Thus, neither Upton, Bhat, or Bhatia, alone or in combination, teach or suggest a token "wherein the token contains user credentials encoded as a platform and application independent string data type" as required by claim 1 of the pending application.

III. Upton, Bhat, and Bhatia do not disclose, teach or suggest a security application program interface and an application program interface coupled to a client application.

Claim 1 (as previously presented) recites, in part, "a security application program interface and an application program interface coupled to a client application on a first platform, and a distinct server application on a second platform to receive the token from the application program interface, the server application communicating with the authentication authority to validate the token..."

Upton does not disclose, teach or suggest "a platform independent security application program interface and an application program interface coupled to a client application on a first operating system, ... and a server application on a second operating system to receive the token from the application program interface, the server application communicating with the authentication authority to validate the token..." as recited in (previously presented) claim 1. In addressing this feature, the Office Action relied on the following disclosure in Upton to read on claim 1:

For example, Upton, discloses in paragraph [0127], lines 1-7 and 18-30:

> FIG. 3 shows an example of a security architecture that can be used with systems and methods in accordance with embodiments of the invention. As shown therein, clients 302, 304 (which may be either physical hardware clients or software applications) may attempt to access a secured service or resource 306, such as a persistent directory

> server, via a transaction or application server 308. …In any case, the connection attempt is received by the transaction server, often via an initial connection filter 320, and is passed to the security service 322. In accordance with the invention, the security service 322 is the focal point for security determination, including client and user level resource access, authorization, certification, privilege assessment and entitlement determination. Enterprise Java Beans (EJB's) 324, Web applications (WebApp's) 326, and other forms of applications may all use the security service through the use of containers. The security service handles calls from these containers to the protected resource, which in the case of FIG. 2 [sic].

Upton also discloses in paragraph [0128], lines 1-10 and 12-18:

> FIG. 4 illustrates an embodiment of the security service architecture 400 in greater detail. The security service augments the basic security services and features provided by the standard Java2 Enterprise Edition security set. As shown in this example, the basic java security set 402 includes security provider interfaces [SPI's] 404 for key storage, authentication, certificate validation, and secure sockets, among others. Customer applications 406 may be written to directly take advantage of the Java security layer and these SPI's. …In accordance with the invention, customer applications are deployed in containers, for example, an EJB container 408 of a WebApp container 410. The containers communicate directly with the security service 414 (herein the same as security service 322), which in turn communicates with the Java security layer 402 and its security SPI's 404.

As shown above and in Figures 3 and 4, Upton teaches a client 302, 304 that communicates with a transaction server 308 to access a secured resource 306. In particular, Upton discloses that the client 302, 304 may attempt to access the secured resource 306 through the use of a security service, and a Java security set including security provider interfaces (SPI's) on the transaction server 308. Therefore, Upton

may disclose performing a secure transaction with a secure resource 306 with a client 302, 304, a transaction server 308, and the secure resource 306, each of which may be implemented on a separate physical computer. However, nothing in this disclosure or elsewhere in Upton disclose a client application having two application program interfaces: a security application program interface and an application program interface. According to the specification of the pending application, the two Application Program Interfaces (APIs) provide several advantages. For example, with the two APIs handling the communication of security information, the client would not need to be aware that a token has been added to an invocation it has sent and a server would not need to be aware that an invocation it has received includes a token. Thus, client applications and server applications could operate in their normal manner without the need for any modifications. (*See, e.g.*, Application at 11, ¶ [0037]).

Additionally, neither Bhata nor Bhatia disclose, teach or suggest "a security application program interface and an application program interface coupled to a client application on a first operating system, ...and a server application on a second operating system to receive the token from the application program interface, the server application communicating with the authentication authority to validate the token," as recited in claim 1.

For at least the reasons established above in sections I-III, Applicants respectfully submit that independent claim 1 is not taught or suggested by Upton. Neither Bhat nor Bhatia, alone or in combination, cure the deficiencies of Upton. Accordingly, Applicants respectfully request allowance of this claim.

Claims depending from Claim 1:

Claims 2-8 were rejected under 35 USC § 103(a) as being unpatentable over Upton in view of Bhat further in view of Bhatia.

Dependent claims 2-8 depend directly or indirectly from independent claim 1 and incorporate all of the limitations thereof. Accordingly, for at least the reasons established in sections I-III above, Applicants respectfully submit that claims 2-8 are not taught or suggested by Upton. Bhat and Bhatia, alone or in combination, do not cure the deficiencies of Upton. Therefore, Applicants respectfully request allowance of these claims.

Claim 9:

Claim 9 was rejected under 35 USC § 103(a) as being unpatentable over Upton in view of Bhat further in view of Bhatia.

Claim 9 includes limitations substantially similar to the limitations discussed in sections I-III above. For example, claim 9 recites, "a method for providing application-to-application enterprise security for different applications on different platforms where there is no continuing context or session and a new context is created with new invocations from one of the applications to another." Claim 9 also recites, "the token contains user credentials encoded as a platform and application independent string data type." Claim 9 further recites, "coupling a security application program interface and an application program interface to a client application on a first platform ... providing, by the application program interface coupled to the client application on the first platform, the token to a server application, the server application on a second platform." Accordingly, the arguments of sections I-III are hereby repeated for claim 9.

For at least the reasons established above in sections I-III, Applicants respectfully submit that independent claim 9 is not taught or suggested by Upton. Bhat and Bhatia, alone or in combination, do not cure the deficiencies of Upton. Therefore, Applicants respectfully request allowance of this claim.

Claims Depending from Claim 9:

Dependent claims 11-27 were rejected under 35 USC § 103(a) as being unpatentable over Upton in view of Bhat further in view of Bhatia.

Dependent claims 11-27 depend directly or indirectly from independent claim 9 and incorporate all of the limitations thereof. Accordingly, for at least the reasons established in sections I-III above, Applicants respectfully submit that claims 11-27 are not taught or suggested by Upton. Bhat or Bhatia, alone or in combination, do not cure the deficiencies of Upton. Therefore, Applicants respectfully request allowance of these claims.

Claim 28:

Claim 28 was rejected under 35 USC § 103(a) as being unpatentable over Upton in view of Bhat in view of Bhatia.

Claim 28 includes limitations substantially similar to the limitations discussed in sections I-III above. For example, claim 28 recites, "a system to provide application-to-application enterprise security for different applications on different platforms where there is no continuing context or session and a new context is created with new invocations from one of the applications to another." Claim 28 also recites, "wherein the token contains user credentials encoded as a platform and application independent

string data type." Claim 28 further recites, "a first security application program interface coupled to the first application on the first platform ... a second application program interface coupled to a second application on a second platform; a second security application program interface coupled to the second application on the second platform, to provide a second security credential." Accordingly, the arguments of sections I-III are hereby repeated for claim 28.

IV. Upton, Bhat, and Bhatia do not teach or suggest two tokens, one generated for the first application and one generated for the second application.

Claim 28, as amended, recites "wherein the tokens generated by the authentication authority are further defined as a first token generated by the authentication authority for the first application based on the first security credential and a second token generated by the authentication authority for the second application based on the second security credential." This element was previously presented in claim 29 which is now canceled. In addressing this element, the Office Action stated that claim 29 recites "the limitations of claims 1-3, 8-9 and 28, therefore, they are rejected applying as above rejecting claims 1-3, 8-9 and 28." (See, Office Action, page 9). However, claims 1-3, 8-9 and the previously presented claim 28 did not recite a first and a second token as the currently amended claim 28 does. Furthermore, Applicants submit that neither Upton, Bhat, nor Bhatia teach the limitation of "wherein the tokens generated by the authentication authority are further defined as a first token generated by the authentication authority for the first application based on the first security credential and a second token generated by the authentication authority for the second application based on the second security credential" as required by claim 28.

For at least the reasons established above in sections I-IV Applicants respectfully submit that independent claim 28 is not taught or suggested by Upton. Bhat or Bhatia do not cure the deficiencies of Upton. Therefore, Applicants respectfully request allowance of this claim.

Claims Depending from Claim 28:

Claims 29-33 were also rejected under 35 USC § 103(a) as being unpatentable over Upton in view of Bhat further in view of Bhatia.

Dependent claim 29 is canceled herein. Dependent claims 30-33 depend directly or indirectly from independent claim 28 and incorporate all of the limitations thereof. Accordingly, for at least the reasons established in sections I-IV above, Applicants respectfully submit that claims 30-33 are not taught or suggested by Upton. Bhat or Bhatia do not cure the deficiencies of Upton. Therefore, Applicants respectfully request allowance of these claims.

Conclusion

Applicants respectfully submit that the pending application is in condition for allowance for the reasons stated above. If the Examiner has any questions or comments or otherwise feels it would be helpful in expediting the application, the Examiner is encouraged to telephone the undersigned at (972) 731-2288.

The Commissioner is hereby authorized to charge payment of any further fees associated with any of the foregoing papers submitted herewith, or to credit any overpayment thereof, to Deposit Account No. 21-0765, Sprint.

Respectfully submitted,

Date: June 15, 2009

/Michael W. Piper/
Michael W. Piper
Reg. No. 39,800

CONLEY ROSE, P.C.
5601 Granite Parkway, Suite 750
Plano, Texas 75024
(972) 731-2288
(972) 731-2289 (facsimile)

ATTORNEY FOR APPLICANTS